Le mécanisme d’héritage

# Démonstration 3 bis du module 3

|  |
| --- |
| Cette démonstration expose la manière de définir une hiérarchie de classes pour mettre en évidence les mécanismes d’héritage |

Déroulement

* Démarrez votre IDE (WebStorm dans les vidéos de démonstration).

Ecrivez le code du fichier HTML nommé « LHeritageEnJavascriptAvecClasses.html » :

<!DOCTYPE **html**>  
<**html lang="en"**>  
<**head**>  
 <**meta charset="UTF-8"**>  
 <**title**>L'héritage en Javascript avec Classes</**title**>  
 <**script src="LHeritageEnJavascriptAvecClasses.js"**></**script**>  
</**head**>  
<**body**>  
  
</**body**>  
</**html**>

* Ecrivez dans le même répertoire le code du fichier nommé « LHeritageEnJavascriptAvecClasses.js » :
* */\*\*  
   \* Class Sport mère  
   \*/***class** Sport{  
   constructor(nom, description){  
   **this**.nom = nom || **""**;  
   **this**.description = description || **""**;  
   }  
    
   afficher(){  
   **return this**.nom + **" "** + **this**.description;  
   }  
  }  
    
  */\*\*  
   \* Class SportDeCompétition fille  
   \*/***class** SportDeCompetition **extends** Sport{  
   constructor(nom, description, niveau){  
   **super**(nom, description);  
   **this**.niveau = niveau || **""**;  
   }  
    
   afficher() {  
   *//Appel de la fonction du parent* **return super**.afficher()+ **" "** + **this**.niveau;  
   }  
  }  
    
  **var** sportDeCompetition = **new** SportDeCompetition(**"Tennis"**, **"Sport de raquette"**,**"Regionale"**);  
    
  console.log(sportDeCompetition.afficher());  
    
  *//Modification du parent*Sport.prototype.jouer = **function** () {  
   console.log(**"Allez-y, jouez!!"**);  
  };  
    
  sportDeCompetition.jouer();  
    
  *//Substitution de fonction*SportDeCompetition.prototype.jouer = **function**(){console.log(**"C'est parti!!"**)};  
    
  sportDeCompetition.jouer();
* Exécutez le fichier HTML dans votre navigateur préféré (n’oubliez pas d’afficher la console) pour observer le résultat.